



UNITED STATES AIR FORCE RESEARCH LABORATORY

Development of an Air Force Wing Level Logistics Cluster for use with the Advanced Logistics Project Architecture

Christopher S. Allen

Patrick K. Clark

Nicholas J. Stute

TASC, Inc.

2555 University Blvd.

Fairborn, Ohio 45324

Christopher K. Curtis
Air Force Research Laboratory

March 1999

Interim Report for the Period February 1998 to March 1999

20000509 114

Human Effectiveness Directorate
Deployment and Sustainment Division
Logistics Readiness Branch
2698 G Street
Wright-Patterson AFB OH 45433-7604

Approved for public release; distribution is unlimited.

NOTICES

When US Government drawings, specifications, or other data are used for any purpose other than a definitely related Government procurement operation, the Government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise, as in any manner, licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use or sell any patented invention that may in any way be related thereto.

Please do not request copies of this report from the Air Force Research Laboratory. Additional copies may be purchased from:

National Technical Information Service
5285 Port Royal Road
Springfield VA 22161

Federal Government agencies and their contractors registered with Defense Technical Information Center should direct requests for copies of this report to:

Defense Technical Information Center
8725 John J. Kingman Rd., STE 0944
Ft Belvoir VA 22060-6218

DISCLAIMER

This Technical Report is published as received and has not been edited by the Air Force Research Laboratory, Human Effectiveness Directorate.

TECHNICAL REVIEW AND APPROVAL

AFRL-HE-WP-TR-1999-0225

This report has been reviewed by the Office of Public Affairs (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER



ALBERT S. TORIGIAN, Lt Col, USAF,
Deployment and Sustainment Division
Air Force Research Laboratory

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
<p>Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.</p>			
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED	
	March 1999	Interim - February 1998 - March 1999	
4. TITLE AND SUBTITLE		5. FUNDING NUMBERS	
Development of an Air Force Wing Level Logistics Cluster for use with the Advanced Logistics Project Architecture		C: F41624-97-D-5002 PE: 63106F PR: 2745 TA: 00 WU: 21	
6. AUTHOR(S)		8. PERFORMING ORGANIZATION REPORT NUMBER	
Christopher S. Allen, Patrick K. Clark, Nicholas J. Stute, Christopher K. Curtis			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)		10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
TASC, Inc. 2555 University Boulevard Fairborn, Ohio 45324-6501		AFRL-HE-WP-TR-1999-0225	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		11. SUPPLEMENTARY NOTES	
Air Force Research Laboratory, Human Effectiveness Directorate Deployment and Sustainment Division Air Force Materiel Command Logistics Readiness Branch Wright-Patterson AFB, OH 45433-7604		Christopher K. Curtis, AFRL/HESR Program Monitor	
12a. DISTRIBUTION AVAILABILITY STATEMENT		12b. DISTRIBUTION CODE	
Approved for public release; distribution is unlimited.			
13. ABSTRACT (Maximum 200 words)			
The purpose of this task was to build a wing-level "cluster society" demonstration software based on the Defense Research Projects Agency (DARPA) Advanced Logistics Project (ALP) architecture. The demonstration software constructed modeled some logistics functions needed to support the deployment of an Air Expeditionary Force. It was concluded from this task that the ALP architecture is a viable solution for building distributed logistic based information systems.			
14. SUBJECT TERMS		15. NUMBER OF PAGES	
ALP AEF Java		52	
		16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

THIS PAGE LEFT INTENTIONALLY BLANK

PREFACE

This technical report contains the results of the Wing-Level Cluster Development task. This task is Delivery Order 23 of the Logistics Technology Research Support (LTRS) program (Contract F41624-97-D-5002). The work described in this report was performed during the period 2 April 1998 through 2 February 1999. The objective of this task was to investigate applying the Defense Advanced Research Projects Agency's (DARPA) Advanced Logistics Project (ALP) architecture to model a subset of the logistics operations needed to support the deployment of an Air Expeditionary Force (AEF). The development of the ALP architecture is being executed by a joint venture between GTE-BBN Technologies and Ratheon Systems called Advanced Logistics Program Integration and Engineering (ALPINE).

The principal investigators for this effort were Mr. Chris Curtis and Capt. Keith Shaneman from AFRL/HESR, Mr. Nick Stute, Mr. Chris Allen, Mr. Steve Betts, Mr. Tim Bowman, and Mr. Pat Clark from TASC Inc.

TABLE OF CONTENTS

PREFACE.....	iii
LIST OF FIGURES.....	v
LIST OF TABLES.....	vi
Introduction.....	1
Background	1
Scenario	3
ALP Architecture.....	4
Key Features of the ALP Architecture.....	5
ALP Cluster Concepts.....	7
Expander Plugin.....	8
Allocator Plugin.....	9
Assessor Plugin.....	10
Data Plugin.....	10
User Interface (UI) Plugin.....	10
LogPlan	10
Cluster Relationships	11
Putting the Pieces Together.....	12
ALP's Decision-Making Philosophy	13
ALP Development Environment	13
The ALP 1998 Society	14
The ALP 1998 – Air Force Standalone Society	16
Air Force Standalone Society – Task Flow.....	18
Penalty Values in the Air Force Society Demonstration	20
Special Supply Requests in the Air Force Clusters	21
WRM Assets Visibility	21
User Interface Components	22
AFFOR/LRC Cluster User Interface	23
AFFOR/LRC Resource View Detail.....	24
AFFOR/LRC Common Support Picture View.....	28
AFFOR/LRC Mobility Processing Chalk Flow View.....	29
AFFOR/LRC LogPlan	31
Fighter Wing Cluster User Interface.....	32
Aircraft Icon View	33
Aircraft Tabular View	34
Assets Allocations View	35
Operational Information View.....	36
Requirements Graph View	37
Results and Conclusions.....	40

LIST OF FIGURES

Figure 1: Cluster Components	8
Figure 2: Plugin Operation Flow	12
Figure 3: ALP 1998 Demonstration Society.....	15
Figure 4: Air Force Demonstration Society.....	16
Figure 5: AFFOR/LRC Window	24
Figure 6: Fighter Wing View Tab.....	25
Figure 7: Provisional Wing Tab.....	26
Figure 8: AFFOR/LRC WRM Tab.....	27
Figure 9: LRC Tab	28
Figure 10: Common Support Picture View	29
Figure 11: Mobility Chalk Flow Processing View	30
Figure 12: Chalk Detail.....	30
Figure 13: LogPlan View.....	31
Figure 14: Fighter Wing View.....	32
Figure 15: Aircraft Icon Tab	33
Figure 16: Aircraft Tabular Tab.....	35
Figure 17: Assets/Allocations Tab.....	36
Figure 18: Operation Information.....	37
Figure 19: Deployment Requirements View	38
Figure 20: 20 th Fighter Wing LogPlan	39
Figure 21: 20 th Fighter Wing Chalk Flow.....	39

LIST OF TABLES

Table 1: Fighter Wing Column Name Descriptions	25
Table 2: Provisional Wing Column Name Descriptions.....	26
Table 3: Fighter Wing Column Name Descriptions	27
Table 4: LRC Column Name Descriptions	28

ACRONYMS

AEF	Air Expeditionary Forces
AFB	Air Force Base
AFRL	Air Force Research Laboratory
ALP	Advanced Logistics Program
ALPINE	Advanced Logistics Program Integration and Engineering
AMC	Air Mobility Command
API	Application Program Interface
CAMS	Consolidated Aircraft Maintenance System
CINC	Commander In Chief
DLA	Defense Logistics Agency
DOD	Department Of Defense
FOD	Foreign Object Debris
GUI	Graphical User Interface
J4	Joint Commands Logistics Director
JDK	Java Development Kit
JFC	Java Foundation Classes
JTF	Joint Task Force
LAN	Local Area Network
LEO	Low Earth Orbit
LRC	Logistics Readiness Center
NCA	National Command Authority
NSN	National Stock Number
OPLAN	Operational Plan
SEAD	Suppression of Enemy Air Defenses
TRANSCOM	Transportation Command
UI	User Interface
UIC	Unit Identification Code
USAF	United States Air Force
UTC	Unit Type Code
WAN	Wide Area Network
WRM	War Reserve Materiel

THIS PAGE LEFT INTENTIONALLY BLANK

Introduction

The purpose of this document is to describe the results for the Wing Level Cluster Development and Demonstration task sponsored by the Logistics Readiness Branch of the Air Force Research Laboratories (AFRL/HESR). This effort involved the evaluation of the distributed communication and workflow architecture involved in the Defense Advanced Research Projects Agency's (DARPA's) Advanced Logistics Project (ALP), as it applies to constructing a wing level logistics information system. DARPA's ALP program is researching ways of using information technology to revolutionize the way logistic functions are performed in planning and execution phases of a military operation. Previous to this effort, the ALP project has focused on architecture implementation and building a sample scenario involving distributed communicating units involved in the planning of a deployment of an U.S. Army Infantry Division. This task represents the first attempt to apply the ALP architecture to the United States Air Force (USAF) deployment planning process.

Background

For several years, AFRL/HESR has been developing technology to improve logistics operations at USAF operational wings. A central goal of this organization's research efforts is to improve the effectiveness and efficiency of logistic processes at the organizational level. This research has ranged from analysis of improved Foreign Object Debris (FOD) control, to technology to provide more environmentally friendly support equipment, to electronic technical orders. Recently, AFRL/HESR has been closely following the availability of more powerful computers, and more robust computer connectivity at USAF airbases worldwide. AFRL has commissioned a number of studies as well as actual prototype development efforts to research ways that logistics information could be made available more quickly and accurately to the people needing it. Having this information in a timely manner improves the ability to accurately perform logistics activities that include planning, packing, shipping persons and materiel to designated operating locations, and keeping those operating locations resupplied efficiently during the execution phase of an operation.

The ALP program is developing a new computer communications infrastructure layer that makes it possible to more rapidly create logistics information systems using a *cluster* architecture. A cluster contains the domain logic required for the logistics operations of an organization. A key to this architecture is the independence of the software application between organizations, so that software can be developed by each organization for its cluster without a tight integration with other clusters representing other organizations. A common application-programming interface (API) between distributed clusters represents the minimum interfacing requirement that must be adhered to in the development of each cluster. This API provides the communications standards upon which clusters communicate logistic requests to each other, as well as the outcome of those requests. The ALP effort has just completed its second year of infrastructure design and development. It has achieved a level of maturity that motivated AFRL/HESR to investigate how this new architecture might be utilized to facilitate the transfer of logistics information, and provide universal access to a variety of tools being developed for wing level support.

AFRL/HESR decided to construct demonstration software compliant to ALP architecture that models a subset of logistic operations at the wing level, as a way of demonstrating ALP capabilities. This demonstration software was intended to support two objectives: (1) demonstrate the feasibility of integrating and providing universal access to existing and future logistics tools by utilizing the architecture; and (2) provide a vehicle to demonstrate these ALP based logistics capabilities at the ALP 1998 cluster society demonstration. To demonstrate the potential capability that an ALP USAF wing cluster would provide it was decided by the government-contractor that a scenario would be needed that involved multiple wing units, as well as other units typically involved in logistics processes. It was additionally decided that it would be desirable to have several wing clusters interact with external organizations, also represented by clusters, in order to highlight the capabilities of the architecture in the ALP 1998 demonstration. The established scenario involved the short-notice deployment of an Air Expeditionary Force (AEF), tasked to fly missions in the theater of operations within 72 hours of notification

to execute a specified operations plan. The following section describes the scenario that was developed.

Scenario

The scenario developed for this effort was a product of iterative brainstorming by the government-contractor team. The Air Force has been organizing AEFs for rapid reaction deployments for several years. This concept, which is still evolving into the Expeditionary Air Force, provides for specific flying units to be maintained at a high state of readiness for immediate employment. To obtain the required responsiveness, information must flow quickly between the AEF, its constituent units, Joint Task Force (JTF) components, in-theater bases, War Reserve Materiel (WRM) locations, Transportation Command (TRANSCOM), and Air Mobility Command (AMC). The following steps approximate the activities that would occur in the future using the ALP architecture during a tasking of the AEF. The demonstration constructed during this effort commences with step 6.

1. National Command Authority (NCA) authorizes the Commander In Chief (CINC) to execute an operational plan (OPLAN).
2. CINC forms JTF.
3. OPLAN formed by JTF.
4. Logistics plan formed by JTF, coordinating with operations plan.
5. Logistics organizations notified of logistics requirements to support OPLAN.
6. One of the top-level logistics organizations notified is the Joint Commands Logistics Director (J4) in the Air Force Forces (AFFOR) component of the JTF.
7. The AFFOR J4 notifies the J4 component of the currently stood up AEF of the start of execution of the specified OPLAN.
8. The AEF notifies its components of their specific missions and asks them to provide their logistics support requirements.
9. AEF units compile their requirements, forward back to the AEF.
10. AEF sends combined requirements to AFFOR/LRC (Logistics Readiness Center).
11. LRC tailors requirements to those really needed.

12. LRC compares requirements of AEF with in theater, near theater, and WRM location assets; AFFOR tailors the requirements and passes the tailored requirements to the AEF.
13. AEF distributes requirements to each of the supporting units.
14. Shortfalls are sourced from alternate units.
15. Multiple sourcing is done on selected "expensive" items, to select the "least expensive" item for transport.
16. Transportation requests for airlift to transport sourced items are sent to TRANSCOM.
17. TRANSCOM provides airlift schedules back to the LRC and to AEF component units.

Upon completing the scenario, the contractor-government team identified the wing organizations that would be represented in the demonstration. For the scenario, it was determined that three squadrons from three separate fighter wings would be deploying to support the AEF. The fighter wings selected include the 1st Fighter Wing, 4th Fighter Wing, and the 20th Fighter Wing. A fictitious provisional wing was constructed to serve as the bed-down location. This provisional wing was given the name of the 123rd Provisional Wing. After defining the scenario and the organizations that would be represented, the team embarked on developing the clusters representing each of the organizations and their associated logic required to ensure sufficient logistics support for completion of their mission. The next section will provide detail on the ALP architecture followed by more specific information on the cluster community (group of communicating clusters) developed during this effort.

ALP Architecture

The general ALP architecture has as its cornerstone the assumption that every organization will have a standard infrastructure for communicating logistics information. Logistics information in this context includes a large amount of messages and data, including requests for equipment, supplies, and ammunition, shipping/airlift schedules, inventory status, logistics readiness status, and airbase bed-down conditions. In the past, such information requests, and replies to those requests, either had to pass through many different information systems or be made by voice communications. The former method,

requiring multiple information systems, is a result of the historical development of systems to solve parts of the logistics information transfer challenge, without adequate consideration of the need to interface with other information systems.

The technology available today has improved to the point that the computers available on logisticians' desktops are capable of performing at a much higher level, and that these computers are in large part now networked to computers around the world via the Defense Data Network and the Internet. In fact, this is the key enabling infrastructure capability that will make ALP processes feasible. With a reliable data communications network in place, much of the information exchange that previously required person-to-person communication can now be done automatically between networked logistics information servers and their clients. Today, these network links are present through high-speed dedicated landlines and telephone networks. In the very near future, high speed capabilities should be widely available even in remote areas through the new constellations of Low Earth Orbit (LEO) satellites being set up by multiple telecommunication providers. Although security issues are still being addressed, it is clear the available bandwidth this technology provides would radically improve Department Of Defense (DOD) information distribution capabilities.

Key Features of the ALP Architecture

The ALP system is a highly automated system. Logistics decision-making logic, the capability to assign equipment and personnel, and the capability to schedule resources and transportation are programmed into the ALP system components. Although automated, users have an important role in the executing ALP system. Users provide the policies and rules that govern these processes, can approve decisions made by the system, intervene when necessary to resolve conflicts, and provide solutions for shortfalls and issues with time constraints. Users can also inject tasks into the system for actions that may not have been considered by the developers, or to incorporate real world events into the processes.

The ALP system is a highly distributed system comprised of many *clusters*. Recall that a cluster is a portion of the “society” representing the domain logic of a particular organization, such as TRANSCOM, Defense Logistic Agency (DLA), or perhaps a maintenance group or a base hospital. The scope of a cluster’s functionality is quite flexible. A cluster’s functionality can be specific to a small sphere of activities or processes, such as the assignment of housing at a base. Or, the functionality may be more encompassing, such as the entire process of deploying a combat mission. The collection of all ALP clusters is called the ALP *society*. A subset of clusters makes up a *community*. For example, the set of clusters that are created to represent the functionality at a Fighter Wing could be considered a community. Note that a given community could have a great deal of processing that is done independently from the ALP society as a whole, e.g., allocating and scheduling resources for training missions, periodic maintenance, food supplies, etc. The same community could also be involved in a society-wide scenario, such as the deployment of forces and equipment for a particular mission.

ALP provides the definition of standard communications protocols that sit on top of the network hardware infrastructure. Clusters provide different types and levels of information, but the ALP architecture ensures that the “clusters talk the same language” so they can exchange information or services.

Based on having a standard set of communication messages between clusters, groups of ALP clusters can be set up to operate with each other, sharing information seamlessly, and requiring very little human intervention to facilitate the communication process. Information, which currently may take several phone call inquiries to different USAF installations, would, in most situations, flow automatically to the clusters requiring it. Automatic updates of data would be provided as real world situations change. The potential seamless and near real time dissemination of messages between clusters could provide the logistician with a more complete up to date picture of the state of an operation or the planning of an operation that is currently available.

The ALP system provides continuous replanning and updating. The status of real world events can change the availability of resources, or can change the priority of future

events. With the ALP system, the availability or status of resources can be monitored and allocated/reallocated to improve timeliness, cost, effectiveness, or to minimize loss of life, etc. These changes can be handled by cluster functionality, resulting in changes to allocated resources, or can cause elements of the cluster's LogPlan to be changed or new elements added. The ALP infrastructure automatically propagates these changes to other clusters, where subsequent plan alterations may be initiated. Moreover, ALP supports the combination of planning and execution requirements. (To date, development has focused upon demonstrating the planning phases, both by ALP engineering efforts and by this AFRL/HESR effort. ALP support for execution events is being addressed, and will be demonstrated, over the next two years of development.) As time passes, planned events become reality, then become things of the past. The results of those events can have an effect on future events. The policies and functionality to support this replanning can be incorporated into the logic of the clusters.

ALP Cluster Concepts

Figure 1 shows the basic components of a cluster. The ALP API defines the methodologies for clusters to communicate with each other and the methodologies for each cluster to communicate information within itself. A cluster makes requests of other clusters via outgoing directives and receives requests from other clusters via incoming directives. Different clusters can be resident on the same machine or reside on different machines. Note that the words "directive" and "task" are used interchangeably in this document and in the ALP architecture documentation.

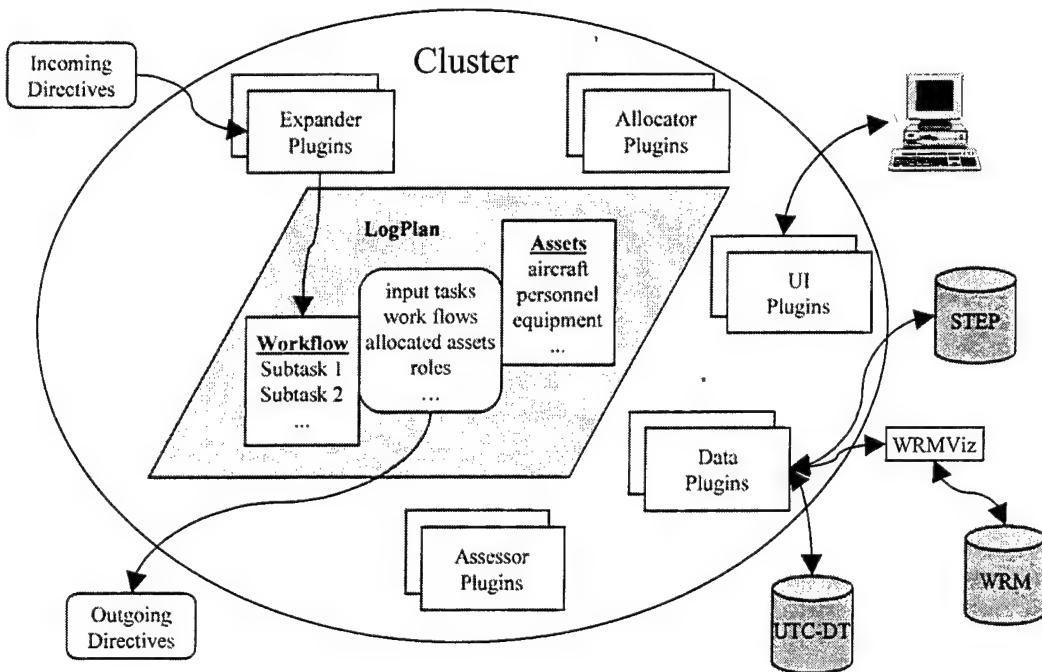


Figure 1: Cluster Components

A cluster consists of ALP LogPlan elements, a combination of various types of *plugins*, and, in order to fit into the society, portions of the ALP infrastructure. Plugins provide the domain specific functionality of the cluster. It is intended that as functionality is created, it can be “plugged into” the cluster, giving the cluster the ability to handle more tasks. As functionality is maintained (fixes, enhancements, etc.), plugins can be “pulled out” and replaced with updated ones, without interrupting the rest of the system’s processing. There are several types of plugins. These include expander, allocator, assessor, data, and UI (user interface) plugins. The following sections provide a description for each of the plugin types.

Expander Plugin

An expander plugin, also called a task expander, performs the initial processing of each input task received by the cluster. This plugin expands input tasks into one or more subtasks that the cluster knows how to complete. Each input task’s set of subtasks is referred to as a *workflow*.

For example, the input task “Generate AEF using OPLAN10A” might be expanded into a workflow containing the following subtasks:

- Subtask 1 = “Determine requirements for Suppression of Enemy Air Defenses (SEAD) using OPLAN10A”
- Subtasks 2 = “Determine requirements for Air Interdiction using OPLAN10A”
- Subtasks 3 = “Determine requirements for Air Superiority using OPLAN10A”

Allocator Plugin

A cluster’s assets can include both physical assets and other cluster assets. It is the responsibility of an allocator plugin to allocate the cluster’s assets to complete the subtasks of each workflow. The allocator plugin may also choose to delegate the responsibility for completing a subtask to another cluster, which is the principal means of creating an output directive. The plugin would choose the target cluster for this kind of directive by means of inter-cluster *relationships* and cluster *capabilities*. These concepts are discussed later in the section titled “Cluster Relationships.”

It is the allocator’s function to maintain the “best” allocation of the cluster’s assets. That is, as the plugin considers assets for new workflows, it may be necessary to reallocate existing asset allocations in order to improve the overall usage of its resources. For example, suppose an allocator plugin has designated a particular truck to perform a transportation request. Further suppose the cluster receives another transportation request. It may be more economical to deallocate the first truck and allocate a single larger one to handle both transportation requests, than to allocate a second truck dedicated solely to the new transportation request. This type of logic would have to be built into the allocator plugin of the cluster.

In addition to allocating the cluster’s assets, it is the allocator’s responsibility to assign schedules of usage and *penalty values* for its allocations. Penalty values represent the costs associated with each allocation, where a cost could be in terms of dollar value, some risk value, a hardship factor for giving up the asset, etc. The penalty value may also be a combination of these things. The ALP architecture developers have not thoroughly

explored the possibilities for the use of penalty values or effective mechanisms to communicate them; the next phase of development will address these issues.

Assessor Plugin

An assessor plugin is responsible for monitoring the execution of the plan. By considering real world events, including satisfactory completion of projected plan components, as well as verifying overall objective conformance, an assessor can watch for plan deviations. The plugin may incorporate various thresholds to ensure that successful plan execution is not jeopardized. Assessor plugins can generate exceptions to alert appropriate mechanisms that remedial actions may be required or may simply insert new tasks into the system to directly effect replanning processes.

Data Plugin

Data plugins are responsible for mapping contemporary data into the ALP society. This is the means for providing an ALP wrapper for existing data sources. It is the data plugin's responsibility to maintain interfaces with its data sources and to act as a liaison between ALP processes and the processes that natively work with each of its data sources. This could include updates to contemporary databases due to ALP processing or may include updates to ALP processing due to triggers set up in the databases.

User Interface (UI) Plugin

UI plugins are created to provide the users with views of ALP's decision making processes and the plan elements that ALP generates. UI plugins also provide the user with the ability to approve or modify the decisions that have been automated by ALP processes. Where applicable, the UI can allow the user to enter or modify rules that govern actions and decisions for plugin processing.

LogPlan

Consider the section in Figure 1 labeled "LogPlan." Each ALP cluster contains a LogPlan that reflects the processes and planning that is accomplished by that cluster. In actuality, this is the collection of all of the cluster's data including, among other things, the

cluster's assets, input tasks, workflows, its relationships with other clusters, and the actual logistics plan elements. Note that each cluster in the ALP society maintains its own LogPlan. The ALP development team is currently working out the architecture and methodology for communicating plan elements to centralized locations in order to generate consolidated plans. The LogPlan maintains the state of a cluster. Currently, the architecture does not provide any persistent storage of the LogPlan. The capability to persistently store LogPlans is being addressed during the 1999 ALP architecture development.

All of the plugin types communicate directly with the cluster's LogPlan. Depending on the situation, this link may be for read-only functionality. For example, certain UI plugins may be created to display various views of the logistics information, such as asset usage, or schedules. On the other hand, when the expander plugin creates a workflow during a task expansion, it writes the workflow directly to the LogPlan. The allocator plugin finds new workflows and available assets in the LogPlan and submits asset allocations, schedules, and penalty values back to the LogPlan.

Cluster Relationships

Clusters form *relationships* with each other, establishing *capabilities* and *roles* in the process. At a minimum, each cluster is required to have a “superior” cluster. The exception to this is the one cluster that resides at the top of the cluster hierarchy. During the cluster's startup phase, ALP establishes a Superior/Subordinate relationship between the cluster being initialized and its superior. Then, while the clusters are processing, each one will have a reference to the other in its list of Organization assets. Moreover, there will be a role associated with the reference, in this case, either “superior” or “subordinate.” In addition to this automatically generated relationship, clusters can selectively establish relationships and roles with each other. These relationships can also include capabilities. For example, cluster A may have cluster B as a “supporting” cluster with “division supply provider” capability. In this example, B has a role of “supporting” cluster A, and A sees B as having the capability of “division supply provider.” When cluster A's allocator is assigning resources to subtasks, it might decide to redirect

subtasks to cluster B when a “division supply provider” allocation would be an appropriate.

Putting the Pieces Together

Figure 2 gives a linear presentation of plugin activity as a cluster and its plugins process a single task. After a cluster receives a task, it is passed to the expander plugin. The expander creates a workflow of subtasks and submits it to the LogPlan. The allocator plugin gets the workflow of subtasks, finds assets to allocate to them (or perhaps assigns tasks to another cluster), calculates penalty values, and submits the results to the LogPlan. The ALP infrastructure creates notification information to pass back to the cluster that made the original request. An assessor plugin may also review the results of the allocations and generate additional directives.

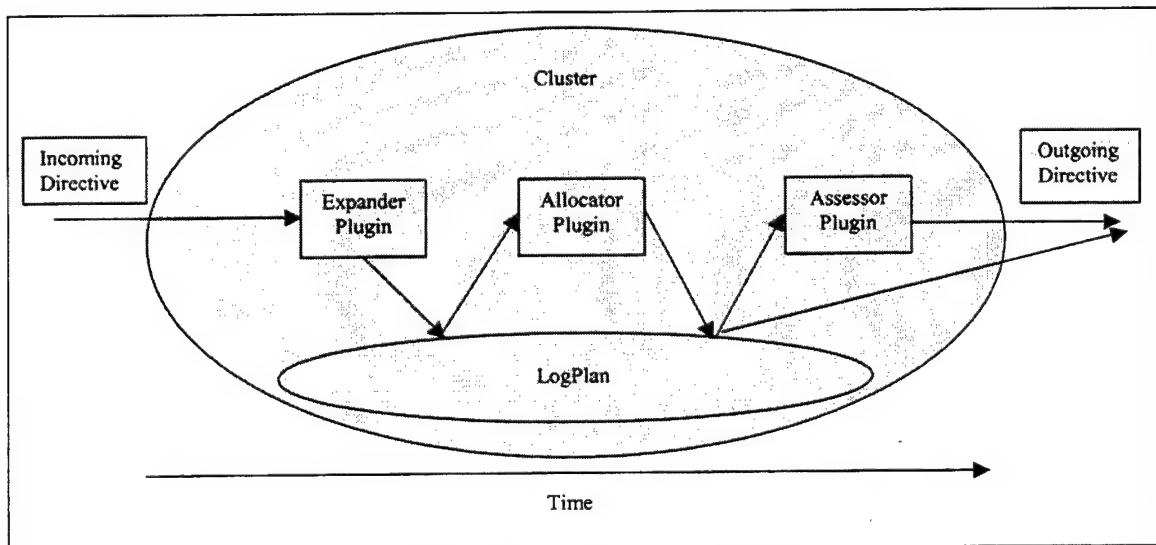


Figure 2: Plugin Operation Flow

ALP's Decision-Making Philosophy

The ALP development process is based on a decision-making philosophy focused on providing solutions that continually improve tolerances rather than attempting to initially provide a “best” solution. The motivation for this is the highly distributed nature of the ALP architecture. To achieve a “best” solution would require a centralized location to request everything of every provider, then decide for everyone, who gets what and when. But, in a highly distributed system, a centralized location containing all the rules and logic, does not exist. Even if it did, the amount of data required to be passed would be prohibitive. ALP provides a different solution. Each task is created with an associated *penalty function*; a penalty function can be thought of as a set of thresholds. Recall from the previous section that an allocator supplies a penalty value when it allocates a resource. The requester of the task will get that allocation’s penalty value and will pass it to the task’s penalty function. If the penalty function indicates the penalty value is “acceptable,” the cluster could just accept the situation and continue. If it is “unacceptable,” the cluster could rescind the task and request a different cluster. This assumes there are other clusters available to do the task; otherwise, “unacceptable” may necessarily be accepted. If the penalty function indicates a “borderline” condition, the cluster could keep the allocation but start creating additional tasks to do some comparative shopping. If the cluster finds a more acceptable allocation from a different source, it could keep the alternate allocation and rescind the original request.

Note that this methodology focuses on keeping all of the elements of a plan within tolerance levels. This approach vastly reduces the number of requests that have to be passed from cluster to cluster, resulting in less burden placed on the communications processes. Also, note that this is where assessor plugins can play an important role. These plugins could generate low-priority requests that are intended to find alternative solutions to improve tolerance levels, but could be processed during “lower” activity times.

ALP Development Environment

The ALP architecture development team elected to use Java and Java-based tools for the development of the ALP architecture. Java provides the platform independence, and

includes powerful networking capabilities as a part of the language. The Java-based tools that ALP has employed include Voyager Version 2.0 from ObjectSpace, Inc., for network protocol communications, POET 5.0 and Oracle for database support, and JGL Version 3.1 from ObjectSpace, Inc., as a standard set of Java utility classes. The current release of the ALP architecture utilizes Sun's Java Development Kit (JDK) version 1.1.6.

The development team for this effort utilized Inprise's Java development environment named JBuilder 2 for constructing the demonstration software. Pentium II based personal computers using Microsoft's Windows NT operating system were the development machines used. Although a personal computer/Windows configuration was utilized, cross platform capability was accomplished as a result of doing all development utilizing the Java programming language. The demonstration software was successfully executed on a Sparc Workstation running the Solaris operating system.

The ALP 1998 Society

As was mentioned in the background section, one goal of this effort was to showcase the AFRL/HESR sponsored demonstration software in DARPA's ALP 1998 Demonstration. The purpose of the ALP 1998 Demonstration was to introduce and provide an update on the state of the ALP architecture, as well as demonstrate the architecture to high-ranking members of DOD. Figure 3 shows the complete ALP society that was created for DARPA's ALP 1998 Demonstration (grouped into communities). The clusters developed under this task were collectively referred to as the "AFDeployed" cluster collection for the ALP 1998 Demonstration.

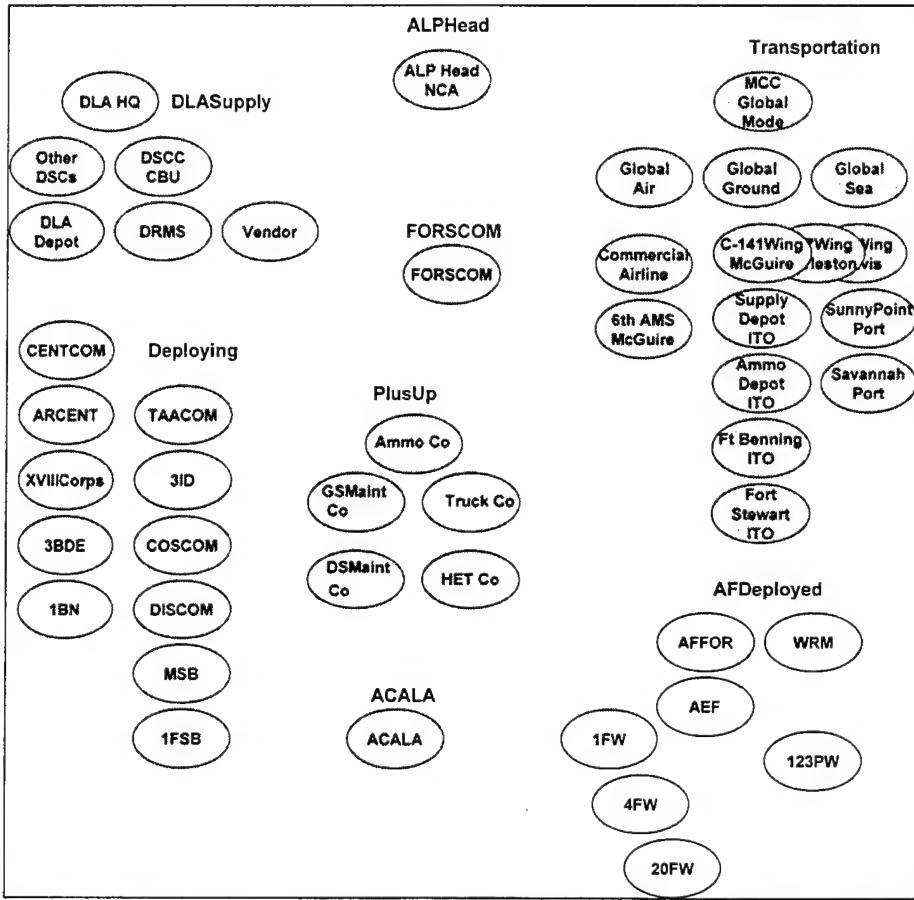


Figure 3: ALP 1998 Demonstration Society

The demonstration successfully simulates the simultaneous deployment of an U.S. Army Infantry Division and an USAF Expeditionary Force. In addition to the community-specific plans being generated, society-wide plan elements were demonstrated, since each deployment generates transportation requests causing the juggling of transportation resources and schedules.

The ALP 1998 Demonstration located the clusters on numerous machines, with several of the machines located at remote sites. This was done to demonstrate the inherently highly distributed nature of the ALP architecture.

The ALP 1998 – Air Force Standalone Society

The AFRL/HESR effort created a standalone society of clusters to demonstrate the generation of an Air Expeditionary Force. Figure 4 shows the clusters that are included in the demonstration. The Air Force clusters are the same in the standalone version as in the full DARPA society described, above with the exception that a considerable amount of functionality was added to the clusters between the November 1998 DARPA deliverable and the February 1999 AFRL/HESR deliverable. One addition included the interaction with the WRMViz database. A stub cluster was created for the standalone society to emulate the TRANSCOM community, and non-cluster functionality was developed, called CMD, for the sole purposes of launching the AEF generation scenario.

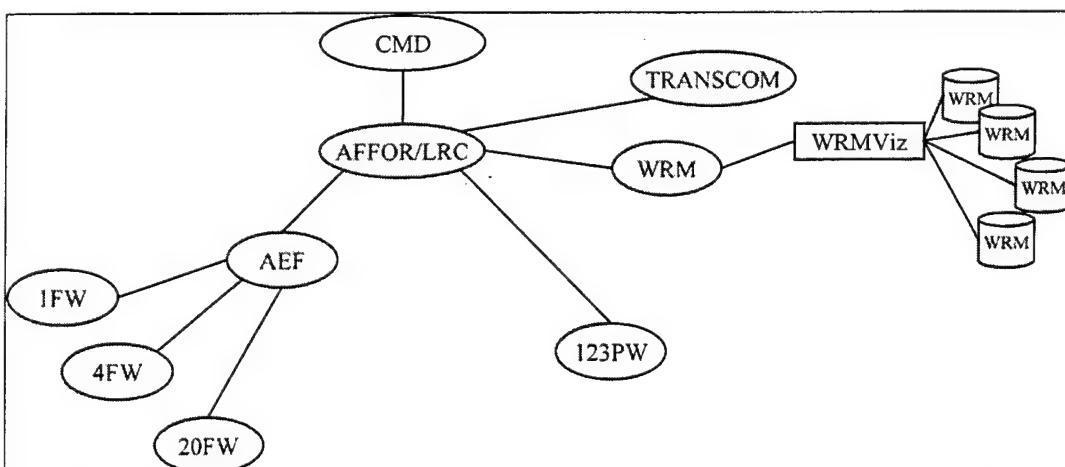


Figure 4: Air Force Demonstration Society

The society depicted in Figure 4 shows the use of a remote data source, the WRMViz utility that was created during a different AFRL/HESR task. The WRMViz utility is described later in the document in the section titled “WRM Asset Visibility.” Under this effort, an API was created to access the WRMViz data sources. This interface was used by the WRM cluster to determine the availability of resources at WRM locations for allocation to the AEF missions and bed-down location requirements.

The 1FW, 4FW, and 20FW clusters contain the domain logic for the deployment of a specific squadron at each of the fighter wings. Note that the goal of this year’s development was for proof of concept; the demonstration is not intended to represent the

entire functionality that would be required to successfully plan an actual mission. Also, note that the Unit Type Codes (UTC) used in the demonstration itemized mission requirements at the pallet level verses itemizing at the National Stock Number (NSN) level. An actual UTC was obtained from Shaw Air Force Base (AFB) and is used by the 20FW cluster to generate requirements for a SEAD mission. The demonstration used simulated data for the UTCs for Air Interdiction and Air Superiority missions, tasked to 1FW and 4FW, respectively. An AFRL follow-on effort will acquire actual UTCs for all the mission types and will refine the granularity of the items down to the NSN level rather than the pallet level. The current implementation houses the UTC information in flat files. The next AFRL effort will convert this information to Oracle databases and will begin the development of more robust and flexible interfaces to these data sources, perhaps via the development of data plugins. This should ease the transition to live data sources in the future.

The 123PW cluster is the bed-down location for the AEF missions. This cluster generates the requirements for such a provisional wing, allocates available resources to satisfy these requirements, and creates output tasks to request the supply of shortfalls items. The 123PW cluster will itself be tasked with numerous supply requests for the items required for the various AEF missions.

The AEF cluster is focused on consolidating the requirements, requests, and efforts of the FW clusters, and providing mediation between the FW clusters and the AFFOR/LRC cluster.

The AFFOR/LRC cluster combines the functionality of what will eventually be two independent clusters. The scenario for this year's demonstration did not warrant the need for separate clusters, one or the other would have contained only pass-through tasks. A pass-through cluster merely passes the tasks it receives on to another cluster. Early versions of the ALP architecture suffered severe performance problems as more clusters were added to the society, and the decision was made to combine the functionality of pass-through clusters into one of its surrounding clusters. Fortunately, the number of clusters no longer appears to have a detrimental effect on ALP's performance.

Air Force Standalone Society – Task Flow

The ALP infrastructure defines a pseudo-grammar to be used to construct tasks. This pseudo-grammar consists of objects, verbs, direct objects and prepositional phrases. Although the various components of each statement are stored separately in Java class elements, the flow of tasks can be described using simple prose. Following is a description of the flow of tasks created during the demonstration. Each task will be highlighted and numbered for easier cross-referencing.

The CMD cluster launches the demonstration scenario by sending the task **1. Generate AEF** to the AFFOR cluster. The expander plugin in the AFFOR cluster is activated since a new input task is received. From the input task 1, AFFOR's expander creates a workflow containing two subtasks: **2. Execute OPLAN10A to 123PW via AEF**, which it sends to the AEF cluster, and **3. DetermineRequirements**

SupportForProvisionalWing for PWforAEF using OPLAN10A, which it sends to the 123PW cluster. Subtask 3 uses nomenclature that is certainly not normal English, but is typical in communicating requests between ALP clusters. In this subtask, “DetermineRequirements” acts as the verb, “SupportForProvisionalWing” is the object that needs a determination of requirements, “PWforAEF” is a specific classification of provisional wing support, and “OPLAN10A” supplies a further refinement of the rules for the determination. Note that the AFFOR cluster could not just make up such a request and expect the 123PW cluster to magically know how to act upon it. Rather, each cluster, the 123PW in this example, publishes the formats of statements that it knows how to process. Then, clusters can create statements that conform to that format and can send them to the cluster that published them.

The AEF cluster receives task 2 and its expander generates **4. DetermineRequirements AirInterdiction to 123PW using OPLAN10A**, which it sends to the 1FW cluster, **5. DetermineRequirements AirSuperiority to 123PW using OPLAN10A**, to the 4FW cluster, and **6. DetermineRequirements SEAD to 123PW using OPLAN10A**, to the 20FW cluster.

The 123PW cluster receives task 3 and, after allocating resources to be ready to serve as the AEF bed-down location, its expander generates the subtask **7. MissionRequirements Manifest to 123PW for PWforAEF using OPLAN10A ofType BaseSupportEquipment**, which it sends back to the AFFOR cluster. The manifest in the request contains the list of equipment required of the bed-down location that the 123PW cluster was unable to supply.

The 1FW cluster receives task 4 and its expander generates the subtask **8. UIRequest_AircraftList AirInterdiction to 123PW using OPLAN10A**. This is a unique request in that the Air Force Society Demonstration incorporates input from the user, and the verb “UIRequest_AircraftList” is a directive created to be handled by a user interface event. In this case, the scenario halts until the user selects and approves a deployment package. Then, the UI plugin gathers the deployed aircraft and generates the task **9. UIResponse_AircraftList Manifest for AirInterdiction using OPLAN10A**. The 1FW cluster’s expander captures this directive and generates the subtasks **10. SupplyAircraft Manifest for AirInterdiction using OPLAN10A** and **11. CreateUTC Manifest for AirInterdiction using OPLAN10A**.

The manifest in the “SupplyAircraft” task is the list of aircraft that the user wanted deployed for the mission. The 1FW allocator plugin processes this request and creates ALP allocations of those aircraft, submitting them to the LogPlan. For the “CreateUTC” task, two tasks are generated and sent to the AFFOR: **12. MissionRequirements Manifest for AirInterdiction using OPLAN10A ofType DirectCombatSupport** and **13. MissionRequirements Manifest for AirInterdiction using OPLAN10A ofType IndirectCombatSupport**.

A similar sequence of events is processed to handle the Air Superiority and SEAD mission requests in the 4FW and 20FW clusters.

At this point, AFFOR has received seven MissionRequirements requests. The AFFOR cluster consolidates the entire set of assets required to generate the AEF deployment.

Using internal algorithms to tailor the requirements, the AFFOR cluster then begins sending supply requests to the clusters to obtain the necessary allocations. There are more than two thousand of these requests propagated through the system during the demonstration.

A similar sequence of events is processed to handle the Air Superiority and SEAD mission requests in the 4FW and 20FW clusters.

When all of the supply requests have responses, (i.e., an allocated resource or a shortfall condition), the AFFOR then sends directives to each of the supplying clusters to generate their transportation requirement requests.

All of the transportation requests are funneled through the AFFOR cluster to the TRANSCOM cluster, which allocates transportation aircraft to satisfy the transportation requirements of the three fighter wings and the WRM.

Penalty Values in the Air Force Society Demonstration

Five of the clusters in the demonstration are used as resources to supply the assets required of the SEAD, Air Interdiction and Air Superiority missions as well as the assets required to supplement the bed-down location. These cluster are 1FW, 4FW, 20FW, WRM, and 123PW. As these clusters allocate assets to satisfy requests, they must calculate and assign a penalty value.

The penalty value assignment is intended to capture a transportation factor and an operational factor. Assets at the 123PW provisional wing have a transportation factor of zero and WRM assets have an operation factor of zero. As a result, the demonstration scenario ultimately sources each asset from the provisional wing until the operational cost surpasses the transportation costs at the WRM. Then WRM sources assets as long as it has assets available. Finally, the fighter wing clusters source the assets. Note that this makes the (perhaps-oversimplified) assumption that WRM sources have a lower transportation penalty value (cost) than those from the fighter wing sources.

Special Supply Requests in the Air Force Clusters

A special directive, called MSupply (multiple supply), was created in the development of the Air Force clusters. The MSupply task is used to make a multiple source request. That is, numerous suppliers can simultaneously be asked to supply an item. The implementation of MSupply automatically compares the penalty values that are returned from the sourcing clusters, keeps the best allocation and rescinds the others.

The MSupply methodology strays from the ALP decision-making philosophy that was discussed earlier. In particular, MSupply places an extra burden on ALP's communications mechanisms. Therefore, MSupply should be used sparingly, perhaps reserved for assets with inherently inconsistent penalty value assignments. Despite this recommendation, all of the supply requests in the Air Force demonstration scenario use MSupply. This will be changed to a more judicious implementation in the future. The current implementation, however, provides the "best" end result because all sources were checked for each attempt to source an asset. Although the communications burden prohibits the widespread use of this approach, it will provide an opportunity to benchmark the ALP processes in test scenarios.

WRM Assets Visibility

War Reserve Materiel operating procedures have not been established which would provide logistics planners with effective access to information regarding WRM assets for use in planning processes. This timely intelligence is a basic requirement for efficient and effective planning and execution of operations.

AFRL/HESR and Synergy, Inc. have developed a prototype-planning tool called WRMViz (WRM Assets Visibility). This graphical user interface (GUI) based tool presents the user with a global map and provides the user with the ability to ask for specific asset availability as well as current locations and readiness. WRMViz also allows the user to allocate assets and to revoke these allocations. WRMViz maintains associations between allocations and specific plans, with numerous plans supported by the system simultaneously.

To support the demonstration software, a Java-based API to access and manipulate the WRMViz database was developed. Using the methods provided in the API, the ALP clusters can programmatically obtain information regarding WRM asset availability and can incorporate these assets into the ALP planning process. The API bypasses the WRMViz graphics displays and user interactions by directly affecting the WRMViz planning processes, serving as an automated extension of the Air Force demonstration system. Three primary interface methods were developed to provide the features necessary to accomplish the WRMViz access described above.

The *findAsset* method obtains a list of WRM locations with sufficient quantities of unallocated, serviceable assets that are free during the time requested. As input to this function, the developer supplies the NSN of the desired asset, the required quantity, and the date the assets are needed.

The *allocateAsset* method is used to allocate assets for use by a specific plan. The developer inputs the NSN for the asset, the required quantity, the WRM location to source the assets, the date the assets are needed, the plan to associate with the allocation, and the Unit Identification Code (UIC) of the unit requiring the asset. This method returns the list of allocated assets, designating the serial number of each asset.

The *deallocateAsset* method removes allocations from the WRMViz database making the asset available for other requests. The developer supplies the NSN, plan, UIC, WRM location, and a list of asset serial numbers that are no longer needed as allocations.

The use of this interface to WRM sources is this first implementation by the Air Force cluster society to gain access to assets from an external (non-ALP) resource.

User Interface Components

This section provides descriptions of the user interface components developed for the AFRL/HESR demonstration software. As described earlier in the document, in an actual fielded implementation, only user interface components pertinent to the organization

would be available. For example, the AFFOR/LRC window would not be available at the fighter wings.

In order to provide a visual representation of what was happening in the demonstration, as well as provide the user with the ability to interact with the demonstration, user interfaces were developed for the majority of the Air Force clusters. These included interfaces for the 1FW, 4FW, 20FW, AFFOR/LRC, 123PW, and WRM clusters. As described earlier, UI plugins were developed for each of the clusters, which provided the capability to access elements of the cluster's LogPlan for display purposes.

The following sections provide detailed description and a series of screen captures for the user interface components developed. All of the user interface components were developed utilizing the Java Foundation Classes Swing (JFC/Swing) library classes. One other product called JChart, by XRTGraph, was utilized to develop the charting capabilities.

AFFOR/LRC Cluster User Interface

The AFFOR/LRC user interface, as shown in Figure 5, is intended to provide an overall view of the deployment operation. This view is divided into three sections. The section in the upper left corner details all the information pertaining to the equipment requests and equipment sourcing to support the deployment. The section in the upper-right corner, titled "Common Support Picture," provides a visual representation of where any particular resource is in its transportation route. Finally, the bottom section of the window titled "Mobility Processing Chalk Flow" provides a graph detailing the times required to load, transport, and unload each chalk of equipment being deployed. The following paragraphs provide more detailed descriptions for each section of the AFFOR/LRC user interface.

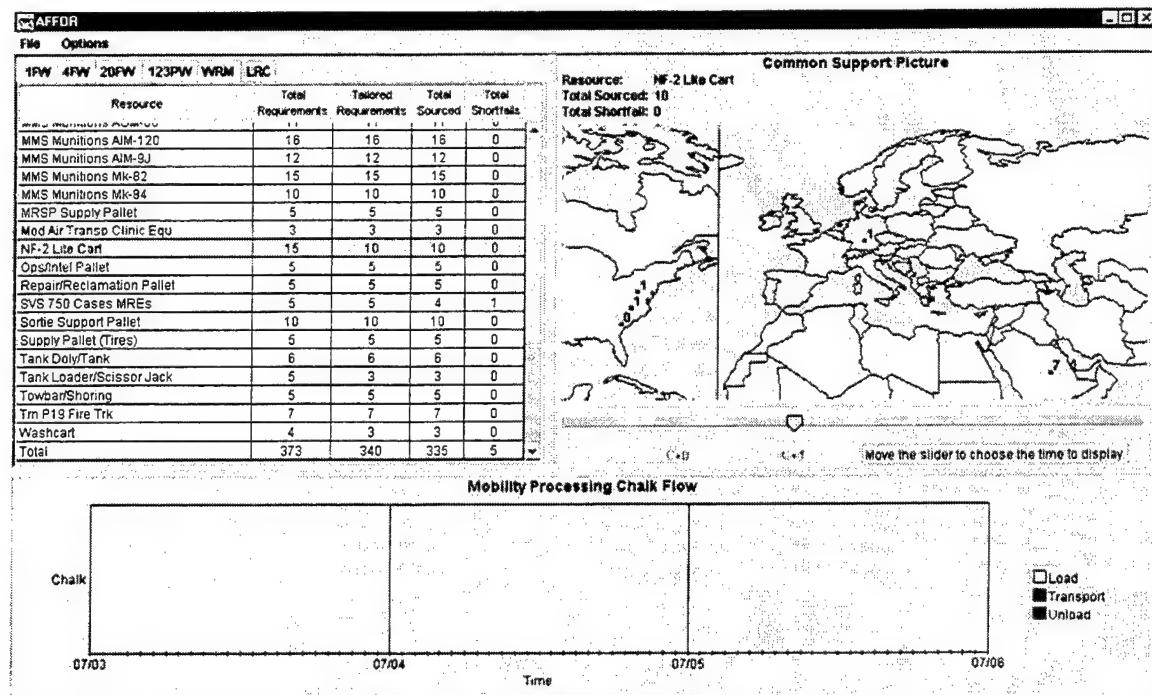


Figure 5: AFFOR/LRC Window

AFFOR/LRC Resource View Detail

The tabular data found in the resource view (upper left-hand section of Figure 5) provides detailed information about the resources being utilized to support the deployment. This view detailed in Figure 6 is divided into six different “tabs” which are labeled: “1FW,” “4FW,” “20FW,” “123PW,” “WRM,” and “LRC.” As each of these tabs is selected, the user is provided with specific information for the selected organization.

The three tabs representing the fighter wings each contain the same set of columns. These columns summarize information about resource requests and allocations from each of the fighter wings. Figure 6 provides a view of the 1FW tab. As can be seen from Figure 6, the column headings for the fighter wing tabs are as follows: “Resource,” “Stand Alone Requirements,” “LRC Requests,” “AEF Allocations,” “Total Sourced,” and “Total Shortfalls.” Table 1 provides descriptions for each of the columns present in the fighter wing tabs.

	1FW	4FW	20FW	123PW	WRM	LCR
Resource	Standalone Requirements	LRC Requests	AEF Allocations	Total Sourced	Total Shortfalls	
APO Forklift 10K All-Terr	0	6	6	6	1	
Aircraft Engine And Engine S...	2	0	0	5	0	
Armament Pallet	2	0	0	5	0	
Armament Trailer	2	0	0	6	0	
B-1 Stand/C-1 Stand	3	0	0	4	0	
B-4 Stand/C-1 Stand	3	1	1	7	0	
Bobtail	2	0	0	4	0	
C-10C Air Conditioner	8	2	2	12	0	
Cabin Pressure Tester	2	0	0	2	0	
Cobra Crane	2	0	0	5	0	
Fuel Systems Pallet	2	1	1	4	1	
Generator M32A-60A	6	0	0	9	0	
H-1 Heater	3	1	1	4	0	
Life Support Pallet	2	0	0	6	0	
MC-1A Compressor	3	0	0	4	0	
MC-2A Air Compressor	4	0	0	8	0	
MC-7 Compressor	3	1	1	4	0	
MHU-141/ECM Pallet	2	0	0	5	0	
MJ-1 Bomblift	3	0	0	4	0	

Figure 6: Fighter Wing View Tab

Table 1: Fighter Wing Column Name Descriptions

Column Name	Description
Resource	Name of the resource
Standalone Requirements	Quantity of the resource needed to support the deployment assuming the fighter wing will be deploying alone
LRC Requests	Quantity of the resource the AFFOR/LRC has requested the fighter wing to supply for the deployment
AEF Allocations	Quantity of the resource the fighter wing is supplying for the deployment
Total Sourced	Total number being supplied by 1FW, 4FW, 20FW, WRM, and 123 PW for a particular resource
Total Shortfalls	Total number that was not supplied

Following the fighter wing tabs, is the “123PW” tab shown in Figure 7. This view provides information about resources being requested and supplied by the 123rd Provisional Wing. The information contained in the 123PW tab is similar to the information found in the fighter wing tabs, except for the second column. Table 2 provides detailed descriptions for each of the columns found on the “123PW” tab.

	1FW	4FW	20FW	123PW	WRM	LRC
Resource	Base Support Requirements	LRC Requests	AEF Allocations	Total Sourced	Total Shortfalls	
APG Forklift 10K All-Terr	7	0	0	6	1	
Aircraft Engine And Engine S...	0	3	3	5	0	
Armament Pallet	0	3	3	5	0	
Armament Trailer	0	3	3	6	0	
B-1 Stand/C-1 Stand	0	1	1	4	0	
B-4 Stand/C-1 Stand	0	2	2	7	0	
BBS 01 Base Power Pro Sys	11	0	0	11	0	
BBS 01 Gen Purpose Fcy Gp	6	0	0	4	2	
BBS 01 Kitchen Mobile Mkt	2	0	0	2	0	
BBS 01 Kitchen-Dining 9-1	5	0	0	5	0	
BBS 01 Remote Area Lite	1	0	0	1	0	
BBS 02 Latrine TT	3	0	0	3	0	
BBS 02 Power 750kW Genera	10	0	0	10	0	
BBS 02 Power Cable Skids	5	0	0	5	0	
BBS 02 Rowpu	3	0	0	3*	0	
BBS 02 SDC-150 KVA	3	0	0	3	0	
BBS 04 Power-60kW Mep-06A	7	0	0	7	0	
BBS 04 Shower/Shave TT	2	0	0	2	0	
BBS 24 Extinguisher Fire	2	0	0	2	0	
TOTALS	100	0	0	100	0	

Figure 7: Provisional Wing Tab

Table 2: Provisional Wing Column Name Descriptions

Column Name	Description
Resource	Name of the resource
Base Support Requirements	Quantity of the resource needed to support the deployment
LRC Requests	Quantity of the resource the AFFOR/LRC has requested the provisional wing to supply for the deployment
AEF Allocations	Quantity of the resource the provisional wing is supplying for the deployment
Total Sourced	Total number being supplied by 1FW, 4FW, 20FW, WRM, and 123 PW for a particular resource
Total Shortfalls	Total number that was not supplied

Following the provisional wing tab, is the “WRM” tab as show in Figure 8. This view summarizes information about resources that the WRM was requested to supply for the deployment. As a result of the WRM being strictly a supply organization, no column representing requirements is present in this view. Table 3 provides detailed descriptions for each of the columns on the “WRM” tab.

Resource	LRC Requests	AEF Allocations	Total Sourced	Total Shortfalls
Aircraft Engine And Engine Support Pallet	2	2	5	0
Armament Pallet	2	2	5	0
Armament Trailer	3	3	6	0
B-1 Stand/C-1 Stand	1	1	4	0
B-4 Stand/C-1 Stand	3	3	7	0
BBS 01 Base Power Pro Sys	11	11	11	0
BBS 02 Power 750KW Genera	10	10	10	0
BBS 02 Power Cable Skids	5	5	5	0
BBS 02 SDC-150 KVA	3	3	3	0
BBS 04 Power-50KVV Mep-06A	7	7	7	0
BBS 9 Air Condg Units	3	3	3	0
Bobtail	1	1	4	0
C-10C Air Conditioner	2	2	12	0
Cobra Crane	2	2	5	0
Generator M32A-60A	4	4	9	0
Life Support Pallet	3	3	6	0
MC-1A Compressor	1	1	4	0
MC-2A Air Compressor	2	2	8	0
MHU-141/ECM Pallet	2	2	5	0

Figure 8: AFFOR/LRC WRM Tab

Table 3: Fighter Wing Column Name Descriptions

Column Name	Description
Resource	Name of the resource
LRC Requests	Quantity of the resource the AFFOR/LRC has requested the WRM to supply for the deployment
AEF Allocations	Quantity of the resource the WRM is supplying for the deployment
Total Sourced	Total number being supplied by 1FW, 4FW, 20FW, WRM, and 123 PW for a particular resource
Total Shortfalls	Total number that was not supplied

The final tab in the resource view of the AFFOR/LRC window is the “LRC” tab shown in Figure 9. This view summarizes information about all of the resources required to support the deployment of the AEF initiated by the AFFOR. As described in an earlier section, the AFFOR/LRC cluster receives deployment requirements from the 1st Fighter Wing, 4th Fighter Wing, 20th Fighter Wing, and 123rd Provisional Wing and applies rules to tailor the requested requirements. The “Tailored Requirements” column of this view identifies for each resource the actual quantity that the AFFOR/LRC cluster requested to be supplied. The amount a particular resource was tailored (reduced from original requests) can be calculated by subtracting the value in the “Total Requirements” column from the value in the “Tailored Requirements” column. Table 4 provides detailed descriptions for each of the columns found on the “LRC” tab.

Resource	Total Requirements	Tailored Requirements	Total Sourced	Total Shortfalls
MMS Munitions AGM-88	11	11	11	0
MMS Munitions AIM-120	16	16	16	0
MMS Munitions AIM-9J	12	12	12	0
MMS Munitions Mk-82	15	15	15	0
MMS Munitions Mk-84	10	10	10	0
MRSP Supply Pallet	5	5	5	0
Med Air Transp Clinic Equ	3	3	3	0
NF-2 Lite Cart	15	10	10	0
Ops/Intel Pallet	5	5	5	0
Repair/Reclamation Pallet	5	5	5	0
SVS 750 Cases MREs	5	5	4	1
Sorbie Support Pallet	10	10	10	0
Supply Pallet (Tires)	5	5	5	0
Tank Dolly/Tank	6	6	6	0
Tank Loader/Scissor Jack	5	3	3	0
Towbar/Shoring	5	5	5	0
Trn P19 Fire Trk	7	7	7	0
Waschart	4	3	3	0
Total	373	340	335	5

Figure 9: LRC Tab

Table 4: LRC Column Name Descriptions

Column Name	Description
Resource	Name of the resource
Total Requirements	Total requirements for a particular resource from 1FW, 4FW, 20FW, 123PW
Tailored Requirements	Total number requested to be supplied of a particular resource
Total Sourced	Total number being supplied by 1FW, 4FW, 20FW, WRM, and 123 PW for a particular resource
Total Shortfalls	Total number that was not supplied

AFFOR/LRC Common Support Picture View

The Common Support Picture view as shown in Figure 10 provides the user insight into the position of resources being deployed to the bed-down location during the deployment. This view works in conjunction with the tabular resource views described in the previous section. The user selects from any tab of the resource view a resource of interest to be displayed on the Common Support Picture. After selecting a resource, the upper section of the Common Support Picture view identifies the selected resource, as well as the total number sourced for the selected resource. Details for the selected resource are also provided on the map.

The map identifies the locations for 1st Fighter Wing, 4th Fighter Wing, 20th Fighter Wing, WRM, and the 123rd Provisional Wing. After a resource is selected, numbers are overlaid on the map for each location providing detail about where the selected resource

is in the deployment timeline. The slider at the bottom of the map depicts the deployment timeline. This slider is divided into 3 24-hour increments where “C+0” indicates the start of the execution of the OPLAN and “C+1” indicates 24 hours past “C+0”. If the user selects “C+0”, the values on the map will identify for the selected resource where that resource is in the transportation route at that time. Similarly, if the user selects “C+1” the numbers will change to indicate where the selected resource is 24 hours past the start of the execution of the OPLAN.

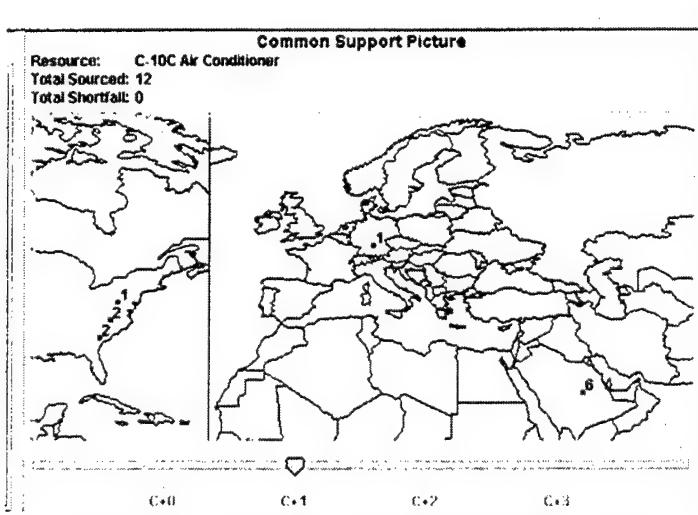


Figure 10: Common Support Picture View

AFFOR/LRC Mobility Processing Chalk Flow View

The final section of the AFFOR/LRC window is the Mobility Processing Chalk Flow view as shown in Figure 11. This view provides timeline details of the movement of cargo from the 1st, 4th, 20th Fighter Wings, and the WRM to the 123rd Provisional Wing. Each entry on the graph represents a schedule for a transportation request. Along the X-axis of the graph is a timeline broken down into hours. The Y-axis identifies each of the chalks of cargo being deployed. Each entry is divided into three different sections. The first section represents the start time and end time for loading the chalk onto the aircraft. The next section represents the transit time of the cargo from the origination point to the final destination. The final section represents the time needed to unload the cargo after landing at the final destination. The user can further drill down into each chalk on this view to get more detailed information.

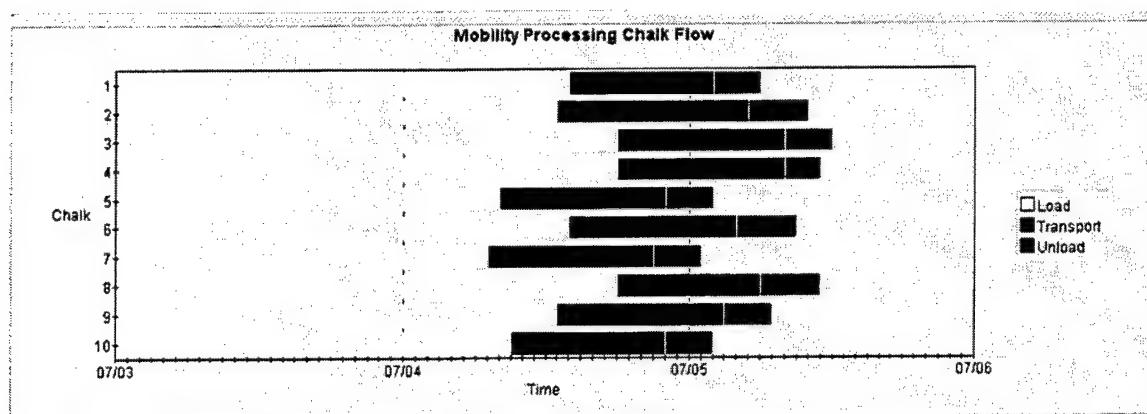


Figure 11: Mobility Chalk Flow Processing View

Figure 12 shows the detail dialog that is displayed when the user selects a chalk from the Mobility Chalk Flow Processing view. This dialog window, titled “Chalk Equipment List,” provides detailed information on a particular chalk. The top section of the dialog lists the chalk number, where the chalk is being transported from and to, and the time the aircraft is estimated to arrive. Additionally, this dialog has a table identifying the items that are being shipped and their associated serial numbers.

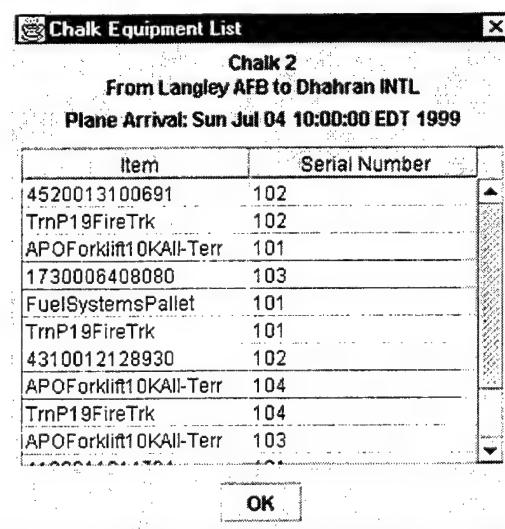


Figure 12: Chalk Detail

AFFOR/LRC LogPlan

Another window that is available from the AFFOR/LRC window is a view of the LogPlan. Details of the LogPlan were defined in an earlier section titled “LogPlan.” This view as shown in Figure 13, provides a tree view of the entire LogPlan for the AFFOR/LRC cluster. This view is primarily used for developers as a debugging tool to confirm that the cluster is receiving the correct incoming directives and that it is sending the correct outgoing directives.

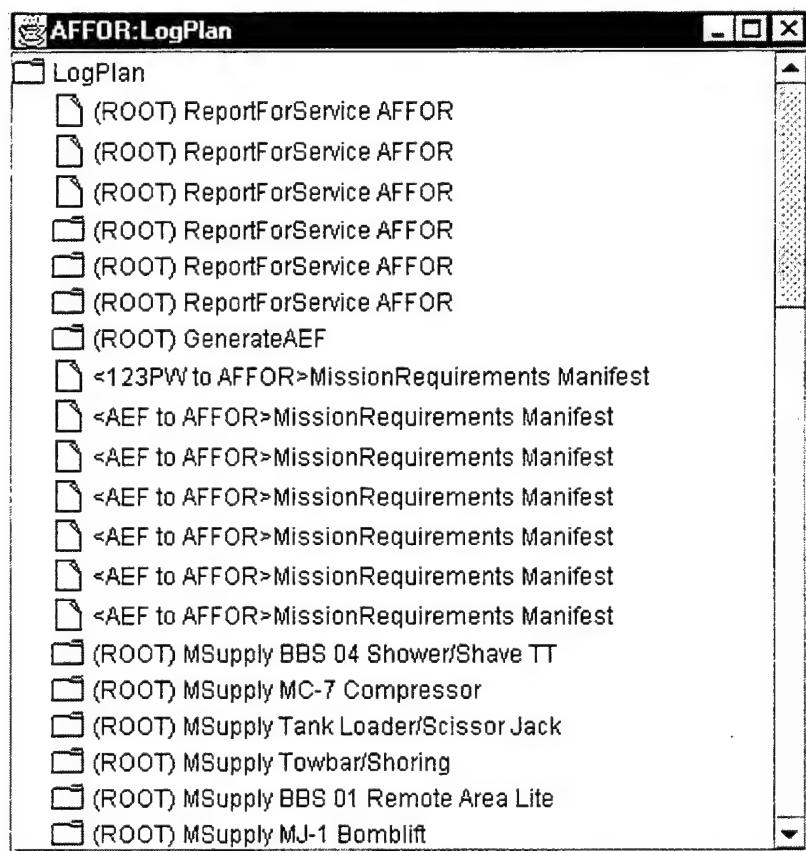


Figure 13: LogPlan View

As can be seen from the above descriptions and screen captures, a substantial amount of information is available from the AFFOR/LRC user interface. The goal of this interface was to provide the user with an overall view of the deployment operation. Much of the design of the AFFOR/LRC user interface was obtained from many meetings with AFRL/HESR personnel, as well as input from TRANSCOM personnel. The next sections provide descriptions of the user interface developed for the three fighter wing clusters.

Fighter Wing Cluster User Interface

The Fighter Wing user interface, as shown in Figure 14, provides detailed information about aircraft status, available assets, and deployment requirements. In addition to providing the user with status information, this window also allows the user to change the deployment package by selecting which aircraft will be utilized to execute the mission. Three main sections exist within this window. On the left side, is a series of three tabbed panes that provide detailed information on the aircraft and non-aircraft assets of the fighter wing. The upper right section identifies information about the operation. Finally, the lower right section contains two graphs that provide detail on the fighter wing's personnel and support equipment deployment requirements. The following sections provide more detailed descriptions and screen captures about each of the three sections. The screen captures used in this section are based on the 20th Fighter Wing, but the same user interfaces are available for both the 1st and 4th Fighter Wings.

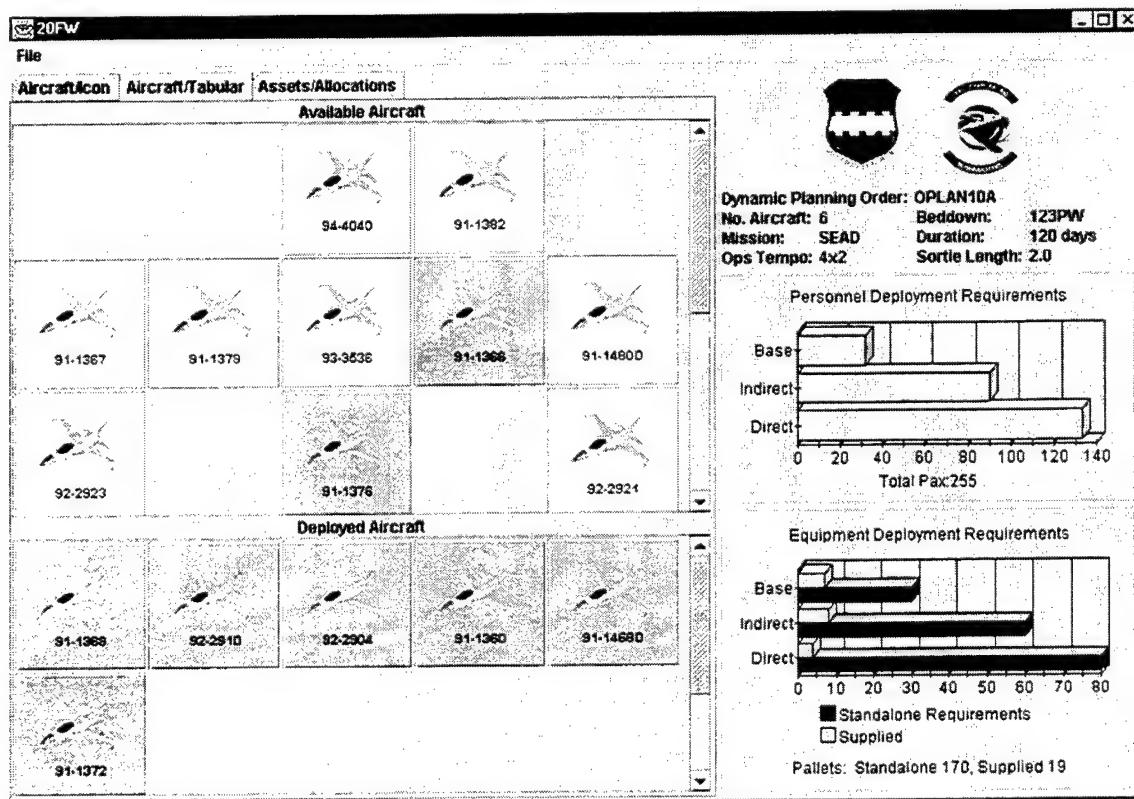


Figure 14: Fighter Wing View

Aircraft Icon View

The aircraft icon view, as shown in Figure 15, provides an interactive view of the fighter wing's aircraft assets. Each aircraft is represented as a square button containing a graphical image of aircraft as well as the aircraft's tail number. The "deployment status" of each aircraft is also depicted on each button by setting the background color of the button. The background is set to green to indicate a "good status," yellow to indicate a "fair status," or red to indicate a "poor status."

This view is separated into an upper and lower section titled "Available Aircraft" and "Deployed Aircraft," respectively. Aircraft appearing in the "Available Aircraft" section are available for deployment and aircraft appearing in the "Deployed Aircraft" section have been selected for the mission. As mentioned earlier, the user has the ability to move the aircraft from the deployment section to the available or from the available section to the deployed section. This functionality allows the user to override the aircraft selected by the computer for the deployment.

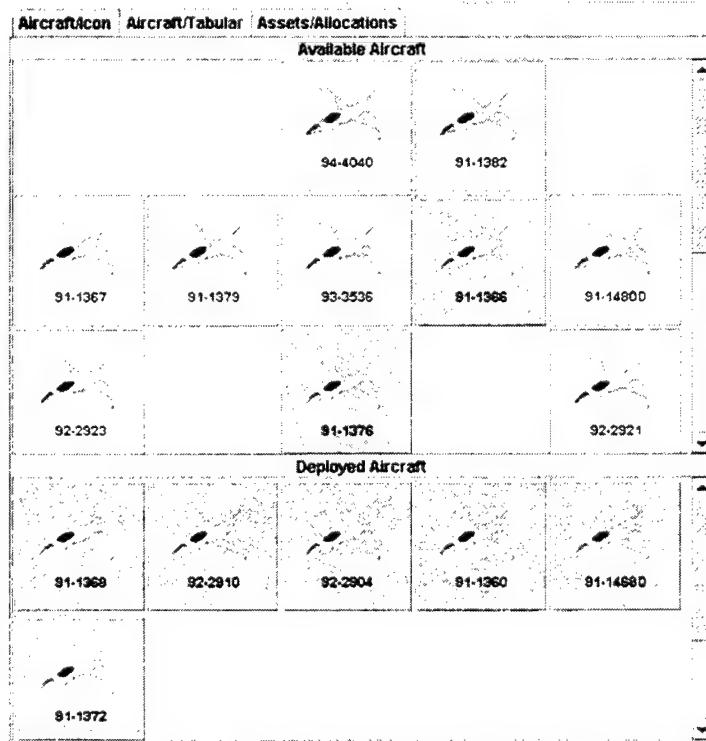


Figure 15: Aircraft Icon Tab

Aircraft Tabular View

The Aircraft Tabular View, as shown in Figure 16, provides another view of the fighter wing's aircraft assets. This view presents a more detailed view of the aircraft than is available from the icon view. As described in the previous section, each aircraft has a status. For this demonstration, three factors were used to determine an aircraft's status. These factors included a projected maintenance score, operational availability score, and mission reliability score for each aircraft. Historical flight and maintenance data was obtained by tail number for the aircraft assets of the 20th Fighter Wing. This data was then applied to formulas developed by the team yielding the three different scores.

The three factors were selected to address both maintenance and operations concerns. When selecting an aircraft for deployment, a maintainer's primary concern is the status of the aircraft's projected maintenance schedule. For example, if an aircraft has an engine replacement scheduled after an additional 30 flight hours, the maintainer might suggest not taking this aircraft for a deployment because of the additional support equipment requirements and time needed for the engine replacement. The other two factors address operational concerns. These factors are the mission reliability and operational availability factors. The mission reliability score calculates how the aircraft fared on each sortie. If the aircraft had anything other than a "Code 1" (no problem reported) return code the score would be lowered. The operational availability score is calculated for each month based on the total fully mission capable hours of the aircraft.

The Aircraft Tabular view consists of table with six columns. The first column identifies if the particular aircraft is selected for the deployment (green check mark) or if the aircraft is available (red x). The next column lists the aircraft tail number. The remaining columns provide the values of the aircraft's projected maintenance, operational availability, mission reliability, and total score.

Another feature available on this view is the ability to change the weights for the projected maintenance, operational availability, and mission reliability scores. By selecting any of these column headings with the right mouse button, the user can select a high, medium, or low weight for the column. This feature allows the user to place higher

or lower importance on each of the three scores depending on the deployment situation. If the user knows a very short deployment is expected, the projected maintenance score may be given a low weight.

Deployed	Tail Number	Projected Maintenance	Operation Availability	Miss Reliability	Total Score
		Weight=Low	Weight=Low	Weight=Low	Med
	91-1368	100	85	77	87
	92-2910	100	83	73	85
X	94-4040	100	68	48	72
X	91-1382	100	80	58	79
	92-2904	100	75	76	84
X	91-1367	100	64	74	79
X	91-1379	100	68	38	69
X	93-3536	100	78	59	79
X	91-1366	100	88	54	81
X	91-1480D	100	76	61	79
X	92-2923	100	72	63	78
	91-1360	100	82	67	83
X	91-1376	100	75	68	81
	91-1468D	100	85	68	84
X	92-2921	100	78	31	70
X	91-1364	100	88	34	74
	91-1372	100	87	62	83
X	91-1398	100	86	55	80

Figure 16: Aircraft Tabular Tab

Assets Allocations View

The final tab on the fighter wing user interface, as shown in Figure 16, is titled “Assets/Allocations View.” This view provides details on the non-aircraft assets of the fighter wing. As can be seen from Figure 17, this view consists of a table with three columns. The first column named “Resource” identifies the name of the resource. The next column named “Quantity” identifies the quantity of a particular resource. The final column, titled “Allocated” lists the quantity of a particular resource the fighter wing has allocated.

Aircraft/Icon	Aircraft/Tabular	Assets/Allocations
Resource	Quantity	Allocated
Aircraft Engine And Engine Support Pallet	3	0
Armament Pallet	3	0
Armament Trailer	3	0
B-1 Stand/C-1 Stand	3	2
B-4 Stand/C-1 Stand	3	0
BBS 01 Remote Area Lite	1	1
BBS 24 Extinguisher Fire	2	2
Bobtail	3	1
C-10C Air Conditioner	4	4
Cabin Pressure Tester	3	1
Cobra Crane	3	0
F16C	18	0
F16Canopy	5	0
F16Engine	5	0
F16Radome	5	0
Fuel Systems Pallet	1	1
Generator M32A-60A	3	2
H-1 Heater	3	1
Life Support Pallet	3	0
MC-1A Compressor	3	1
MC-2A Air Compressor	3	2
MC-7 Compressor	3	0
MHU-141/ECM Pallet	3	0
MJ-1 Bomblift	3	1
MJ-2A Test Stand	3	2
MJ-4 Bomblift	3	1
MMS Munitions Mk-82	1	0
MRSP Supply Pallet	3	0
NF-2 Lite Cart	3	1
Ops/Intel Pallet	3	0

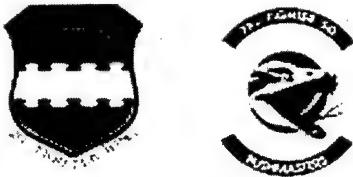
Figure 17: Assets/Allocations Tab

Operational Information View

The upper right section of the fighter wing window, as shown in Figure 18, provides information about the operation and deployment. The information included in this section includes the following:

- Name of the plan
- Number of aircraft needed to support the mission
- Name of the mission
- The operational tempo
- Bed-down location
- Duration of the deployment
- Sortie Length

Additionally, the logos for the particular fighter wing, as well as the squadron are provided on this view.



Dynamic Planning Order: OPLAN10A
No. Aircraft: 6 Beddown: 123PW
Mission: SEAD Duration: 120 days
Ops Tempo: 4x2 Sortie Length: 2.0

Figure 18: Operation Information

Requirements Graph View

The final section of the fighter wing view, as shown in Figure 19, consists of two graphs detailing personnel and equipment deployment requirements. The “Personnel Deployment Requirements” graph shows the personnel requirements for the deployment broken out by direct, indirect, and base personnel. Where direct personnel are personnel needed to directly support the mission, such as pilots and flight line maintenance. Indirect personnel are personnel required to sustain the mission, such as back shop support . Finally, the base personnel are those personnel needed to support base level functions for the mission, such as military police.

The “Equipment Deployment Requirements” graph provides detail on the equipment needed for the deployment. There are two bars for each of the direct, indirect, and base equipment requirements. The top bar indicates the combined requirements and the bottom bar indicates the standalone requirements. The combined requirements bar represents the amount of equipment the fighter wing is providing for the mission. The standalone requirements bar indicates the amount of equipment the fighter wing would supply if it was deploying alone.

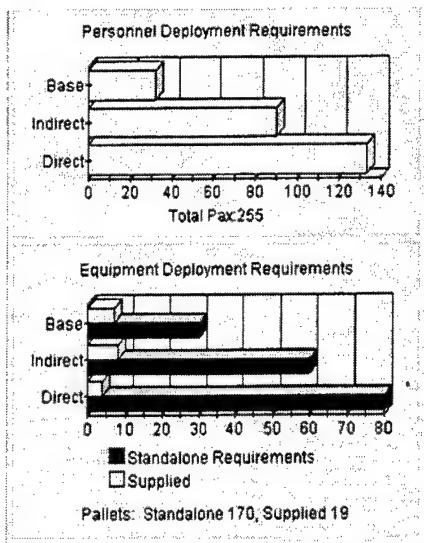


Figure 19: Deployment Requirements View

Two additional windows are available from the fighter wing window's "File" menu. These windows include a view of the fighter wing's LogPlan and a view of the transportation requests made by the fighter wing. The LogPlan view is identical to the LogPlan view described in the "AFFOR/LRC LogPlan" section. Figure 20 shows a view of the 20th Fighter Wing cluster LogPlan.

The transportation requests window available from the fighter wing provides identical information to what is provided in the "Mobility Processing Chalk Flow" view available on the AFFOR/LRC window. A description of the information provided in this view is given in the section "AFFOR/LRC Mobility Processing Chalk Flow View." The only difference between the views is that the chalk flow window available from the fighter wing only shows transportation requests made from the particular fighter wing. Figure 21, shows the chalk flow window for the 20th Fighter Wing cluster.

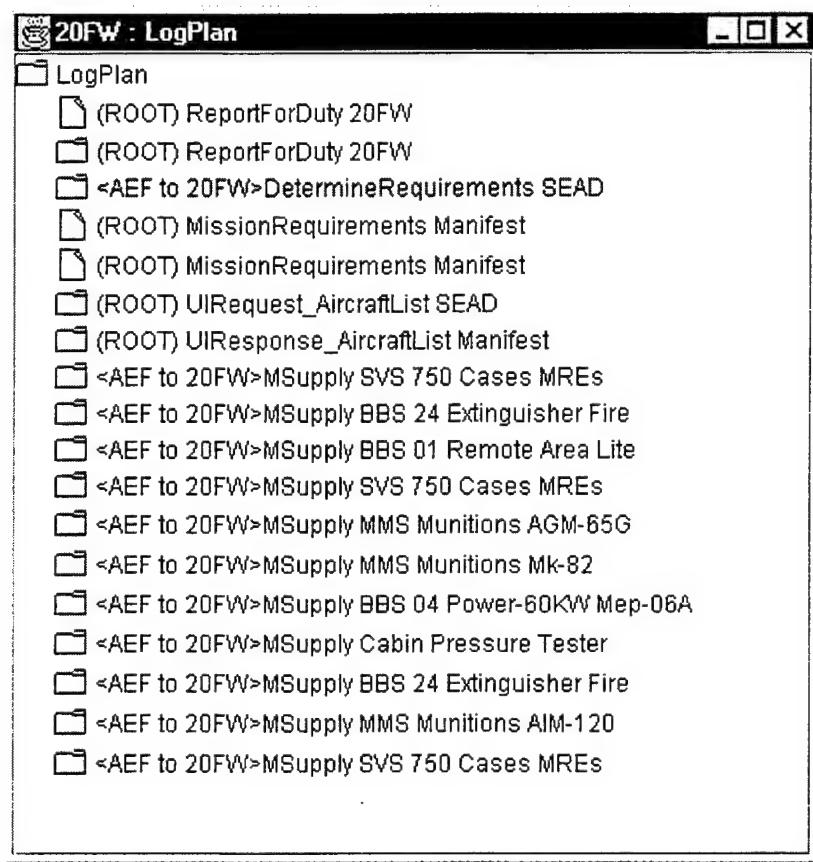


Figure 20: 20th Fighter Wing LogPlan

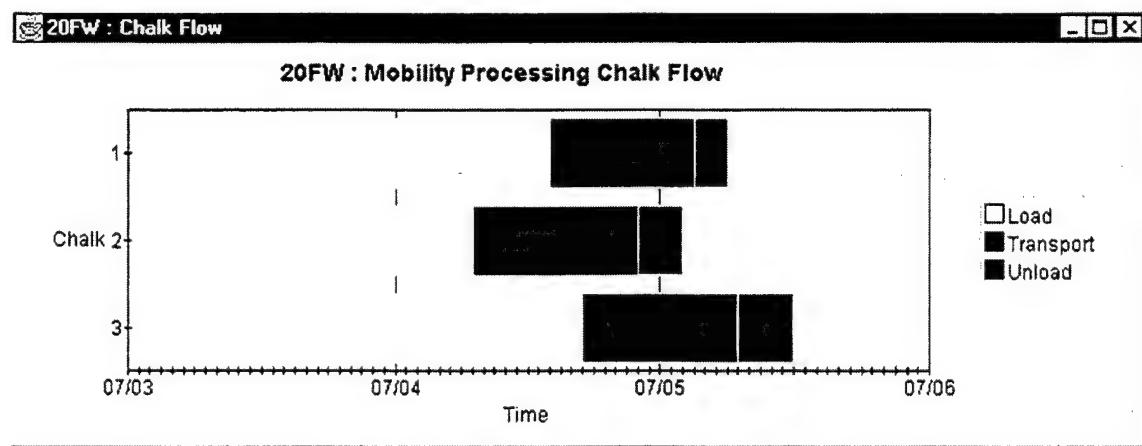


Figure 21: 20th Fighter Wing Chalk Flow

Results and Conclusions

During this task the government/contractor team successfully constructed a cluster society built on the ALP architecture. Several accomplishments of the effort were key to advancing the understanding of ALP and how it might be applicable to USAF logistics. These results included inter cluster communication of tasks, negotiation for lowest cost item among various sources, rescinding tasks, and distribution of clusters across multiple machines on a local area network.

The cluster community developed under this effort provided a wide range of features that were crucial to the evaluation of the ALP architecture. First, the cluster community that was developed implemented a larger number of task types than had previously been implemented; including the passing of user defined prototype instances. Second, the AFRL/Contractor team implemented a much larger number of task instances (over 4000) within our scenario, more, perhaps, than any other community developed for the October 1998 ALP demonstration. Third, the AFRL/Contractor team was the only one to implement a scenario that required and used the penalty and notification functions for the demonstration. Each of these significant extensions of the basic inter cluster communications helped to point out limitations of the current ALP infrastructure software, and, therefore, enabled ALP development team to more rapidly fix or enhance the infrastructure, and have more comprehensive test cases available to verify fixes.

As previously discussed, the scenario involved numerous tasks to flying units (wing clusters) to provide individual items needed for deployment of an AEF. In the government/contractor discussions about the scenario, it seemed imperative that there be the ability to find the "cheaper" alternative source for any given item needed for the deployment. In many cases, the cluster trying to assign the assets may make a "conscious" decision not to "shop around" for the best deal. This may be done based on a rule or set of rules that lead to the conclusion that those items tend to be about the same cost among the sources generally queried, and that the marginally small difference is not worth pursuing. However, the option must exist to be able to have the sourcing cluster look at multiple sources for a critical item, and after evaluation assign the best source to

supply the item. The best source for the item will be the one with the lowest penalty value, where penalty values can be weighted by several different and potentially competing factors including purchase cost, shipping cost, availability date, shipping location, arrival time, batch factors, and others. A relatively simplistic scheme for assigning penalties was developed (this is an area that must be carefully developed in a fielded system) to ensure correct weightings of the factors. However, the team was the only plug-in developer group to actually implement the concept of multiple sourcing. Even with a simplistic penalty function, it was demonstrated that this could be applied to a realistic scenario, and verified that the ALP infrastructure could support the kind of asset "negotiation" that is typically performed by USAF planners today.

Along with the ability of a cluster to ask multiple sources to provide their best cost for a particular item, there is a corresponding need to be able to say "never mind" to clusters that come back with a high cost. Since the ALP infrastructure software currently will cause each cluster requested to supply an item to actually allocate that item, in a multiple source situation, the "losing bidders" must be told to "deallocate." This functionality is accomplished by using the infrastructure capability to rescind tasks. To the development team's knowledge, our team was the only one to exercise this functionality of the infrastructure. Again, this was of great benefit not only because it eventually allowed the demonstration of a more realistic scenario, but also because it provided early diagnosis of problems with the rescind functionality. Our team was instrumental in helping the ALP architecture developers correct the way rescind worked, particularly in passing a rescind request down a chain of previously tasked clusters. This functionality will be very critical in the succeeding years of the ALP program, since, in any real planning system, there will be a need to remove certain tasks due to changes in the real world situation or the operational plan over time.

Finally, the cluster community developed under this effort was tested with sub-communities distributed across a local area network, with as many as three nodes hosting one or more clusters. This demonstration showed the ability of the large number of tasks generated by the community to be communicated across machines. This is a critical scaling issue in moving the ALP architecture out of the laboratory and into the real world.

The premise of the real world operational ALP system is that clusters will be distributed across the global information network, including landline networks, phone dial-ups, and satellite links. While none of these wide area communication paths were tested during this effort, it was a useful first step to test the distribution on a local area network (LAN). As a result of our development, several problems were identified with the infrastructure software and its ability to work in a distributed environment. These problems were resolved and, now that the basic functionality is working, the ALP development effort can concentrate more on improving the efficiency so that wide area network (WAN) scalability can be achieved.

The results of the Wing ALP Cluster Development and Demonstration effort show clearly that there is great promise in the ALP architecture for building a new generation of USAF logistics information systems. The ALP architecture allows clusters to be built to communicate with existing contemporary systems and databases (e.g., Consolidated Aircraft Maintenance System (CAMS)), without redeveloping entire systems. The benefits of the architecture allow an integrated logistics system to be built out of "pluggable" new or existing systems at different command echelons. By complying with the architecture, the integrated system can provide almost seamless transmission of required tasks and data to organizations that need them. The bottlenecks of phone and even E-mail inquiries will be radically reduced if the ALP vision of intelligent decision making clusters is realized. In the relatively simple demonstration developed under this effort, the deployment planning scenario showed the benefits of the automated tasking and resource allocations that could occur between cooperating organizations using the cluster architecture. While there were several iterations on the infrastructure software during this effort, the performance kept improving, and over the next few years, we believe the performance and capabilities of the basic infrastructure will continue to improve. One remaining question is whether the architecture is globally scalable. The answer to this question will be driven not only by logistics data requirements at different levels, but by the evolution of network technology for mobile and deployed units. As the bandwidth increases with these systems, it will become more likely that a scalable system can be built. An important observation to make is that the ALP cluster system does not

have to solve every logistics problem to be a success. Many decisions could be delegated to the automated evaluation of penalty values, while those considered too critical or complex could be left to human decision-makers. There may be a subset of logistics tasks and data that would fit within the globally available bandwidth and still provide significantly advanced planning speed and quality over what is possible with today's systems.

REFERENCES

“ALP PlugIn Developer’s Guide” Version ALP-PDG 3.2, draft rpt. (An ALPINE Joint Venture Document)